Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science 18 (2013) 1505 – 1514

Procedia Computer Science

International Conference on Computational Science, ICCS 2013

Robot Task Allocation using Signal Propagation Model

Małgorzata Żabińska, Tomasz Sośnicki, Wojciech Turek, Krzysztof Cetnarowicz

AGH University of Science and Technology, Krakow, Poland

Abstract

The problem of Multi-Robot Task Allocation has received significant attention over last years. The solutions with decentralized decision making have proven better durability than those using centralized planning. In this paper a method which does not use any explicit communication is presented. It is especially suitable for very high dynamics of tasks priorities, when other methods must often perform time-consuming replanning. The method uses a simple model of signal emission and propagation in the environment. Simulation experiments are provided to demonstrate usability and interesting features of the method.

Keywords: multi-agent systems, discrete simulation, multi-robot task allocation

1. Introduction

The idea of increasing performance and durability of a robotic system by using a group of robots, which can simultaneously solve several tasks, creates a problem of Multi-Robot Task Allocation (MRTA). The problem has received significant attention over last years, which is justified due to complexity and multiplicity of variants. The basic variants of the MRTA problem have been categorized by Gerky and Matarić in [1]. Three features of the problem have been taken under consideration:

- ability of performing just one or multiple tasks simultaneously by a single robot,
- existence of tasks which require one or more than one robot to accomplish,
- static list of tasks vs possibility of new adding new tasks during execution.

The classification of problem variants also provides an extensive survey of solutions, which led the authors to the conclusion that there are still many unsolved problems in the area.

The solutions to the MRTA problem can be categorized, according to task allocation strategy, into three types: centralized, distributed, autonomous.

The centralized solutions use single algorithm executed on a single computer to make all decisions concerning the allocation of the robots to the tasks. Typically this involves selecting a leader of the group [2]. The algorithm needs a knowledge about the environment, the robots and the tasks, which requires intensive communication.

E-mail address: zabinska@agh.edu.pl.

Depending on complexity of the problem (determined by considered variant) different optimization algorithms can be used to calculate the best allocation.

The centralized approach can theoretically give the best (closest to optimal) results. Several issues attributed to this method, including single point of failure, very intensive communication and inability reacting fast for changes, make researchers focus on decentralized approaches.

To overcome these problems many variants of distributed algorithms have been proposed. Typically they are based on a predefined negotiation protocols, which determine what information should be exchanged and how the robots should react in particular situations [3]. This type of solutions does not have a single point of failure, however they still suffer from complex communication protocols and inability to react fast for changes.

Another group of solutions to the MRTA try to solve the allocation task with no explicit communication between robots. Decisions of particular robots are based on observation of the environment and other robot (like for example the low-level task assignment in [4]). This kind of swarm solutions are considered more robust, because functioning of the system is never endangered by a single robot failure. However defining a fully autonomous algorithm for a task as complex as MRTA is not straightforward.

In this paper a fully reactive algorithm for solving the task allocation problem is presented. It does not use any explicit communication between robots. The robots make decisions using observations of a single directional signal, which propagates in the environment. The signal incorporates attractor of tasks and repeller of the other robots. The robot control algorithm makes it possible to look for tasks (perform efficient foraging) and to divide available robots between several tasks.

A model of signal propagation is proposed, which does not require information about the signal source to properly distribute the signal in the environment. It is inspired by natural signals present in the physical world, like sound, smell or light. It can also be implemented using an artificial sensor network. Therefore the algorithm can be very useful for building robust systems solving MRTA problem for highly dynamic problems.

The following section presents the inspiration for the method. Third section provides details of the signal propagation and task allocation algorithm. The fourth section describes the most interesting experiments, which were designed to present interesting features of the approach.

2. Agent-Based Approach to Robot Management

When we introduce a concept of agent to management of robots (or other mobile vehicles) [5, 6, 7], we may consider two kinds of relations between the agent and the robot. In the first approach, the agent is connected with the robot and resides in its computer. It is the classical solution, when agent plays the role of a software controller for the robot. It manages the robot taking into account conditions of surrounding environment, including other robots. The agent can communicate with other agents residing on computers of other robots existing currently in the environment. There also exists possibility of contact of agents with server existing in the cyberspace to use the stored resources. It is related to such resources as information and computing power. As we have already mentioned, it is classical configuration of robot software configuration.

In the second approach the agent related to the given robot, acts in the cyberspace in some virtual environment created there [8, 9]. In this case the real environment is mapped into the cyberspace (with the use of appropriate tools), making the virtual environment being the model of environment from the Real space. Agents are associated with robots acing in the reality. We can go further and consider the agent from the cyberspace as the robot, when the robot in reality makes a tool of associated agent (so now the robot) from the cyberspace. It enables transfer of robots management (partially or even totally) to the cyberspace [11] and the use of tools and methods used for agent systems. Some extension is treating some collaborating agents as a robot. Such a group of agents being the cyberspace robot manages the classical real space robot. Similarly, we can consider the situation when the cyberspace robot has a set of tools (groups of classical robots) in the real space [10] . Such agents may be called AGENT-ROBOTs (AR) and robots in the real space are called TOOL ROBOTs (TR) or classic robots. Robot (R) may be defined with the use of couple $R = (AR, TR)$ (fig. 1).

An example of application of agents acting in the cyberspace to solve “the Task Allocation Problem” may be a system for management of mobile robots group to empty garbage containers in the city environment.A task to

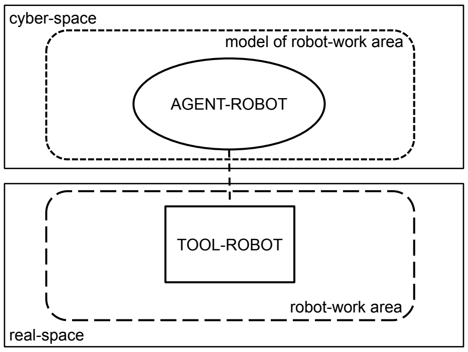

Fig. 1. Schema of the agent in the cyber-space and the robot in the real-space.

manage these robots (garbage trucks) consists in such control of them, that they are directed appropriately to full containers, without loosing time to excessively circle around down the city streets.

The appropriate agent system, functioning in the cyberspace realizes this task.

Let us assume the following model of an urban environment functioning according to the below-mentioned rules and comprising the elements such as network of streets with crossroads, garbage cans (containers) set in the given points (in streets or crossroads), a given number of mobile robots (garbage trucks) circulate in streets. Garbage cans are filled in with waste with unknown speed (tempo impossible to be anticipated). Each can has a sensor, which may transfer the level of its filling in into the computer system in the cyberspace, The role of the circulating robots (garbage trucks) is to empty containers (cans). Emptying operation should take place when the garbage can is full to some extent (to avoid working on empty or almost empty cans), but emptying process should be done with such a frequency to not to overflow the garbage can.

The agent system in the cyberspace has the following structure: streets network is mapped into the graph (streets are represented by edges, whereas crossroads by nodes), in the given points of the graph garbage cans are marked with the information about their state of filling (e.g. in percentage). Each robot is related to an agent, which may move along graph edges. An agent related to the given robot may perform a number of operations such as: it may fix a position of a robot in a city, and map the position into a graph in the cyberspace taking the position on its own what implies that it is able to fix its current position in the whole graph. The agents have may control the corresponding robot, directed it to the appropriate route according to its plans. So the agents task is to direct its own robot to the can, which is to be emptied. Common inspiration to realize the described solution was a concept of pheromones applied to ant systems.

Assuming that the ant system and the agent system are similar, we may use in the proposed agent system principles similar to the concept of pheromones [12], and propose an approach based on smell of garbage stored in garbage cans. Thus undertaking the decision on the route choice, for agents moving in the cyberspace (and mobile robots in reality) is related to smell.

However the disadvantage appears when there are some robots (agents) to empty garbage cans. The filled in can may attract several agents (robots) which reach the container in order to empty it, while only one robot (agent) is necessary. To improve the approach we can take into consideration two kinds of smell:

1. smell of garbage i.e. filled in garbage cans attracts agents-robots (trucks) which are to empty them,
2. smell of agents-robots that is repulsive for the trucks.

It is possible to consider different ways of taking into account given off smells and different ways of undertaking decisions on the basis of smells existing in the environment. The proposed system may be an example of application of management of robots in the real space with the use of simulation of the real environment in the cyberspace using a concept of an agent. The garbage cans may be considered as a problem to be resolved, garbage robots may be considered as robots for resolving problems and emptying process is a task resolving problems by robots.

3. Environment Model and Task Detection

Presented scenario may be generalized into a system composed of a group of robots is used to accomplishing abstract tasks. Each task require one robot. Tasks do appear dynamically in different (randomly chosen) locations of the space. Each task has a priority and the priorities dynamically change.

3.1. System Model

We can consider a system in real-space that is composed of the elements as follows:

- real-space composed of area that is a piece of 2D space,
- mobile robots called Tool-Robot TR existing in the area,
- problems arising in different regions of the area.

Robots are to accomplish tasks that consist of identifying the problem, going up to it and resolving the problem (fig. 2).

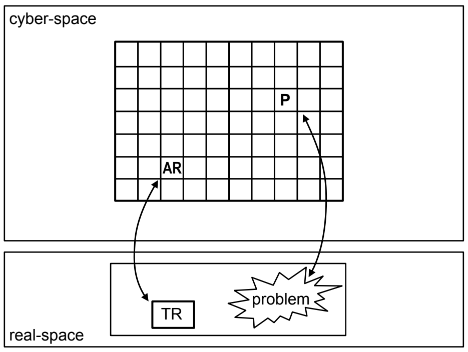

Fig. 2. Schema of a robot and a problem in the real-space and an agent in the cyber-space

The described system is modeled in the cyber-space.

- The 2D area of the real-space is represented by a chessboard composed of $m * n$ cells c_{ij} that form an environment for agents AR and problems P. The discretized environment may be considered as a grid of nodes, where each one is connected with at most 8 closest neighbors.
  In every cell an agent AR may remain or problem P may arise.
- Problems (tasks) that arise in a given locations – determined cells of the chess-board environment. A given problem is characterized by priority which can change over time. The priority is represented by a real value.
- Agent-Robots (AR) that represent Tool-Robots (TR) . Agents AR may move between cells to reach given target (fig. 2). Agent AR is to realize a task and for this purpose it has to identify problem P, find the itinerary to the identified problem and go up to it (what means that the robot resolves the problem).

Every cell $c_{i,j}$ consist of 9 logical subcells $c^{x,y}_{i,j}$ (fig. 3), where $x \in \{N, 0, S\}$, $y \in \{W, 0, E\}$.

$$\begin{aligned} C &= \{c_{i,j} : 1 \leq i \leq m,\ 1 \leq j \leq n\} \\ SC &= \{sc^{x,y}_{i,j} : 1 \leq i \leq m,\ 1 \leq j \leq n,\ \ x \in \{N, 0, S\},\ y \in \{W, 0, E\}\} \end{aligned} \tag{1}$$

Problem P that appear in a given cell $c_{i,j}$ generates a signal $SM^{x,y}_{i,j}$ in all subcells($c^{x,y}_{i,j}$), representing direction of the signal. The intensity (real value) of the generated signal is proportional to the priority of the problem that appeared in the cell. Agents may also generate signal with a given intensity appropriate to this agent. This signal may also propagate among cells of environment. So the signal in a given subcell may be result of generation or propagation process.

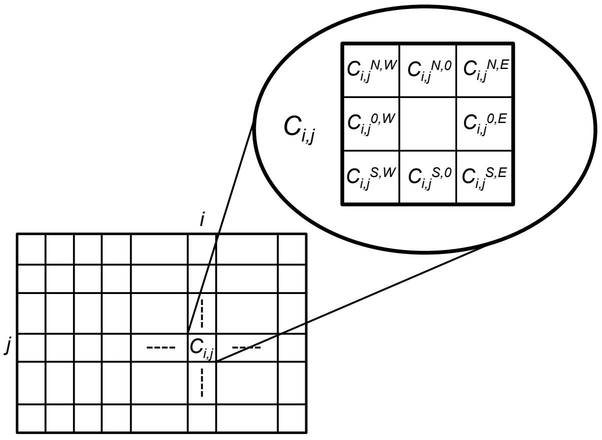

Fig. 3. Schema of the model of the 2D area represented by a chessboard of cells

$$\begin{gathered} SM : SC \to \Re \\ SM(sc_{i,j}^{x,y}) = SM_{i,j}^{x,y} \quad 1 \le i \le m,\ 1 \le j \le n,\ \ x \in \{N, 0, S\},\ y \in \{W, 0, E\} \end{gathered} \tag{2}$$

Function SM defines the current repartition of the signal over the environment SC. The $\mathcal{SM}$ may denote the set of signal repartitions over the environment.

The features of the propagation of the signal $SM_{i,j}$ are as follows:

- We can define a function $PROPAGATE$.

$$\begin{gathered} PROPAGATE : \mathcal{SM} \to \mathcal{SM} \\ PROPAGATE(SM_{i,j}^{x,y}) = SM'^{x,y}_{i,j} \end{gathered} \tag{3}$$

  Function $PROPAGATE$ transforms the intensity of signal ($SM_{i,j}^{x,y}$) in every subcell $c_{i,j}^{x,y}$ of the environment as the result of the propagation of the signal. As the result of the propagation the intensity of the signal in cells of the environment is modified. Particular $PROPAGATE$ function, which has been tested in this research, uses specific propagation rules, which has been shown in fig. 4. The rules guarantee that the signal propagates to all cells without duplications.
- The intensity of the sent signal is supressed while it is sending what is defined by the propagation factor PRF, proportional to the distance between cells.
- We have attenuation of the signal over time, with use of the $ATTENUATE$ function.

$$\begin{gathered} ATTENUATE : \mathcal{SM} \to \mathcal{SM} \\ ATTENUATE(SM_{i,j}^{x,y}) = SM'^{x,y}_{i,j} \end{gathered} \tag{4}$$

The proposed rules define some features of propagation: one signal will visit every node in range only once, propagation takes place without knowledge about source.

3.2. Task Detection

The signal medium SM may be used to navigate agents toward problems P. The navigation process is as follows:

- Agents can move in any direction from one cell to other neighbouring cell, one cell at a simulation step. The destinating cell is selected due to the analysis of the signal intensity in neighbouring subcells.
- Agents are attracted by the signal of a problem P – always moving in the direction of the most intensive signal.

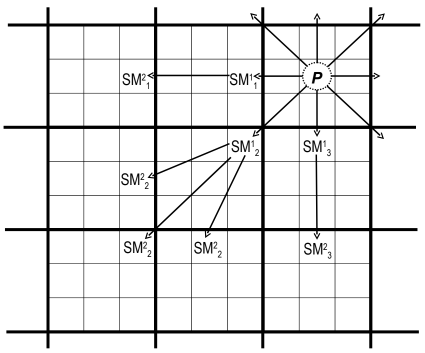

Fig. 4. Schema of the propagation of the signal-medium SM in the environment in cyberspace

- Agents are repulsed by the signal of another agent – always moving in the direction of the least intensive signal.
- If an agent is enabled to find the best neighboring cell using the signal medium (SM) analysis it take direction of displacement randomly.

The agent's itinerary is properly traced if attraction by problems and repulsion by other agents is well-balanced by the agent. Using the described methods of displacements between cells agents may find the best way to the problem.

4. Implementation and Experiments

Simulation framework has been written in C++ and Qt library. The main advantage of this solution is a high-performance of the programming language. The simulator is very flexible. It can simulate a lot of different cases and environments. Configuration of resources, agents, connections between them and the initial events are placed in an XML file. The algorithm of agents, resource properties, event execution logic is written in C++ code and compiled into the DLL library. This approach allows you to easily and quickly change the behaviour of each system component.

Results of simulations can be saved as text or binary files, it depends on a developer. Each component can be drawn on canvas and can be shown in a viewer, it means that the framework allows you to paint the whole environment in particular time. The whole drawing system is made by QPainter class from Qt library.

The major purpose of the experiment is to investigate whether robots with a small amount of information about the environment can effectively find tasks. The tasks included in the environment emit a signal which attracts the robots. Additionally, the robots emit signal (RSM), which repels other robots. The experiment shows the positive influence of the repulsion on the robots distribution in the environment. The time which is needed to full expiration of a signal emitted by a robot is called ATF (Attenuation Factor).

The experiments were divided into three scenarios. The first scenario involves only two robots and two tasks. This simple example shows the repelling effect. The results of the experiments depend on the simulation parameters (e.g. a RSM intensity, attenuation factor (ATF)). Their influence is shown in Scenario 2. Also in this scenario, the quality of a solution was presented. An additional advantage of the repulsion adding is to increase the search area. In the last scenario, the efficiency of task search is shown in two cases – without RSM and with RSM propagation.

The following common assumptions are made for all experiments.

- The number of robots is constant and defined at the beginning of the simulation. Also the start position and the robot speed have to be defined at the start.
- There is the constant number of locations in the simulation. Location is a place (a plane cell), where a task can appear.

- The tasks appear randomly during the simulation. The frequency and an amount of tasks are random numbers limited by parameters defined by the user. A task appearance increase location priority.
- Signal propagation is faster than robots movement. For one robot step, there are five signal moves.

All robots in the simulations are represented by the white square with thick black border. Tasks locations are drawn as black squares. Current value of signals is represented as brightness of square subcells with the following meanings:

- white - no signal or very low signal intensity,
- gray - medium robot or task signal intensity,
- black - high robot or task signal intensity.

4.1. Scenario 1: simple allocation experiment

In this scenario there are two robots and two locations for tasks appearance. The first location is placed in the upper half of a plane and the second location is in the lower half. Robots are placed in top left corner. When the simulation starts, tasks are randomly added to locations and they start to attract the robots.

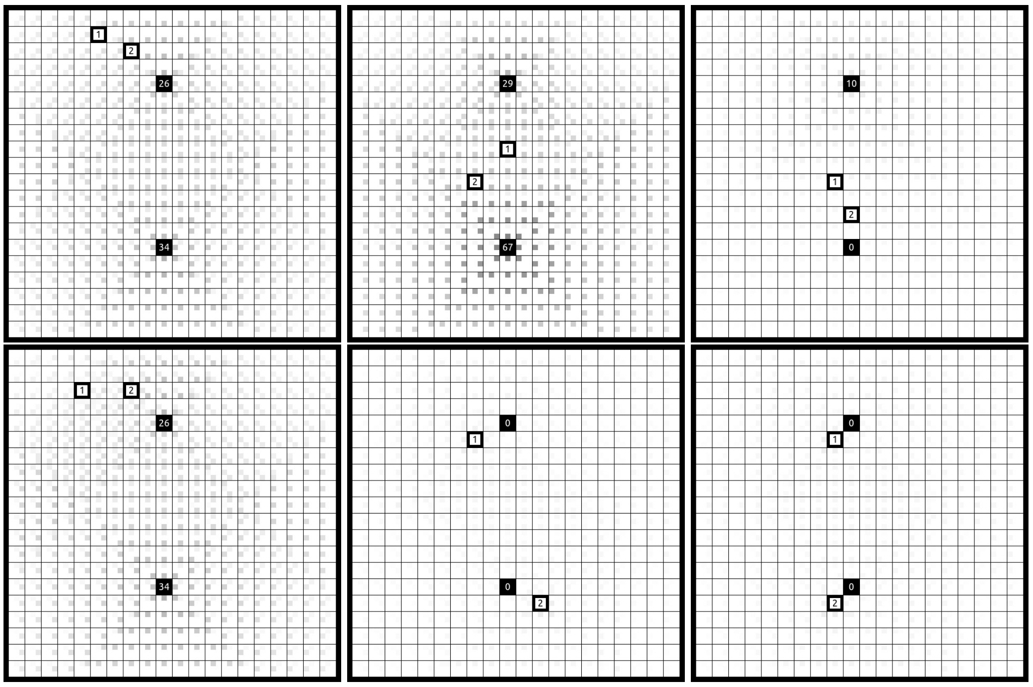

Fig. 5. Comparison of robots movement in a simple allocation experiment. Simulation step 7 (left column), step 100 (middle column) and step 172 (right column). Top: the robots without the RSM, bottom: robots with the RSM.

Images in the left column of figure 5 show the robots behaviour shortly after starting the simulation. The distance between robots is bigger in the case with RSM propagation. The robots in the middle top image can stay close to each other. It is inadvisable behaviour; the robots should rather scatter over the entire plane. When there is no robots repelling, robot can come close to the same task location where another robot is waiting for task (top-right image). It is also unwanted situation. If there is a robot near the location, other robots should not approach to this place.

4.2. Scenario 2: complex allocation experiment

In the second scenario locations with tasks are separated into two groups (clusters) placed in different parts of the plane. At the start of simulation, the robots are located in the center of a plane. In this experiment a random amount of tasks (from 1 to 10) is generated in random intervals (from 1 to 40) to all locations in the plane.

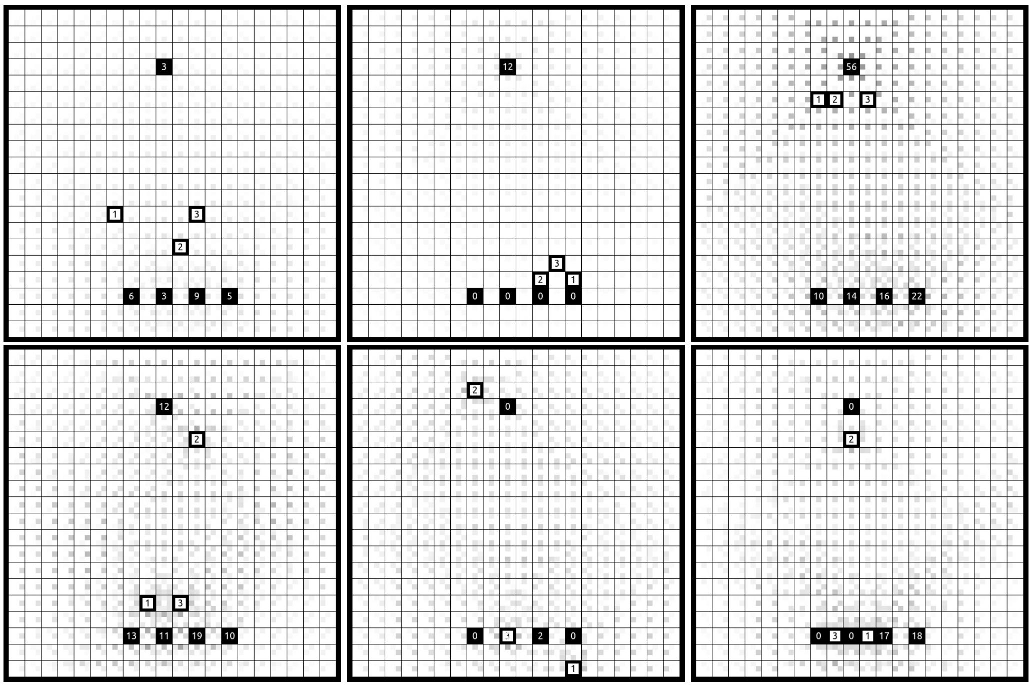

Fig. 6. Comparison of robots movement in a situation with cluster of tasks location. There are two cases - the robots without the RSM (top) and with the RSM (bottom). Simulation step 7 (left column), step 100 (middle column) and step 172 (right column).

Figure 6 shows the tasks location placement and the robots movement. At the beginning of the experiment (left column) there are different robots behaviour. Adding a RSM to a robot (bottom image) separates the group to two parts. It is an excellent situation, because the robots can carry out the tasks simultaneously.

The robots without the RSM move in a group from one location to another. In this case the attractor value of three tasks is too strong to separate robots into two groups. The top-right image presents the situation, where the robots come together to one location instead of take cares of all locations. The experiment with the RMS shows the most important feature of the approach: the robots can automatically split into groups proportional to the attractor of tasks in the environment.

To measure the quality of the solution we used a penalty value proportional to the time needed to fulfill the tasks. The value is defined as follows:

$$Q = T * V \tag{5}$$

where T is a portion of time (dependent on the simulation time resolution) and V is an amount of tasks. The lower value of Q, the better is. In the ideal case, the tasks are immediately taken from the location, so Q is equal to 0.

Figure 7 presents the results of the experiments with different parameters. Each experiment was executed 10 times, average values are presented.

In general, adding a small RSM and ATF always improves the robots behaviour. The tasks wait shorter for the robots. The higher the value of the RSM the more regular robots placements is. However, too high RSM value pushes the robots on the edge of the plane. If the signal is too small, it has not an impact on robots and they move too close to each other. The optimal value of the parameters depends on tasks generation frequency and the size of the environment.

4.3. Scenario 3: foraging experiment

This experiment shows the behaviour of robots at the absence of tasks. Robots search the area to find any task. Without any signal in the environment, they move in random direction. If the RMS is present, very interesting results can be achieved.

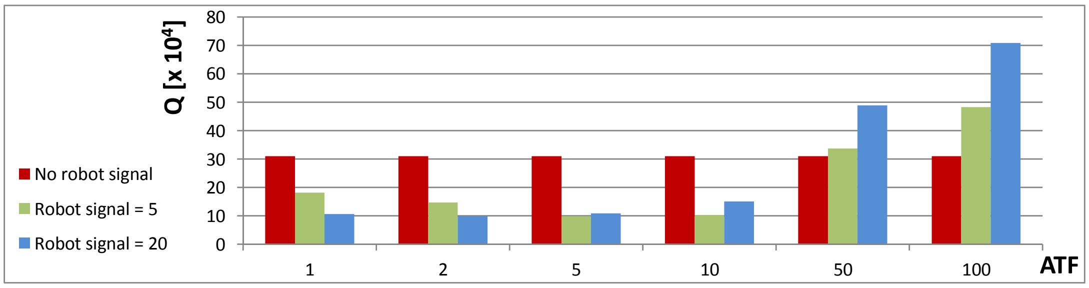

Fig. 7. Quality of solution in respect to different RSM values and a different repellent attenuation factor (ATF).

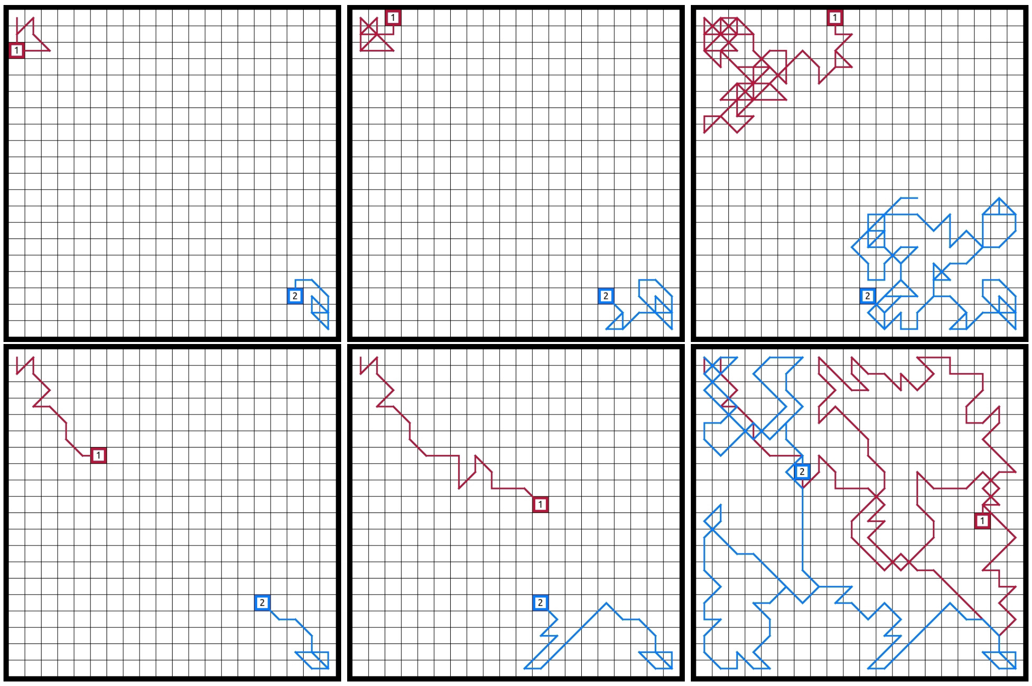

Fig. 8. Comparison of robots movement at the absence of tasks - without the RSM (top) and with the RSM (bottom). Simulation step 10 (left column), step 20 (middle column) and step 100 (right column).

Left images of a figure 8 shows the simulation after first 10 robots moves. The offset of the robots without RSM from the start position is only 2 cells. When the RMS is present, the offset is larger (4 and 6 cells). The effect of a robot being repelled by its own signal appears here. This signal pushes a robot to new areas of the environment.

Next step (middle images) shows the situation after 10 more moves. Without a RSM (top) the robots are still in the corner of the plane. The same simulation step with a robots with a RSM looks better. The robots occur closer to the center of a plane.

After 100 moves, robots without signal emission searched less than half of the plane. They do not leave the quarter of the plane, where they were initially placed. In the case with RSM, the robots covered much bigger part of the plane. They have searched almost the entire plane. This is caused by the repelling of the place already visited. The shape of the paths created by the robots depends on the value of RMS.

4.4. Summary

As the 3 above scenarios show, adding the repelling signal enhances the results. However, if a value of the signal is too high, the results are worse than in the case without a signal. The strong signal pushes the robots to the edge of the plane. When the robots repels each other properly, the average distance to task locations is smaller.

A simulation parameters selection (such as RSM or ATF) depends on a tasks location placement, generation frequency and robots amount. These values can be obtain empirically.

5. Conclusions and Further Work

The solution for the problem of task allocation in multi-robot system presented in this paper has a very interesting and promising features. A very simple model of information propagation was used, which is inspired by natural signals propagation in the physical world. The group of robots was able to successfully distribute in the environment and accomplish a number of dynamically appearing tasks without using explicit communication.

Further research of the presented model is needed in order to solve more complex variants of the MRTA problem. Also the signal distribution function must be extended to accept environments with obstacles.

Acknowledgements

The research leading to this results has received founding from the Polish National Science Centre under the grant no. UMO-2011/01/D/ST6/06146.

References

[1] B. P. Gerkey, M. J. Mataric: A Formal Analysis and Taxonomy of Task Allocation in Multi-Robot Systems. The International Journal of Robotics Research September 2004 23, pp. 939–954.

[2] M. B. Dias, A. Stentz: A Market Approach to Multirobot Coordination, Technical report of The Robotics Institute, Carnegie Mellon University, 2001.

[3] X. Zheng, S. Koenig: K-swaps: Cooperative negotiation for solving task-allocation problems, Proceedings of the Twenty-First International Joint Conference on Artificial Intelligence, 2009, p. 373–378.

[4] D. Zhang, G. Xie, J. Yu, L. Wang. Adaptive task assignment for multiple mobile robots via swarm intelligence approach. Robotics and Autonomous Systems 55, 2007, p. 572–588.

[5] J. Kozlak, J.-C. Creput, V. Hilaire, A. Koukam, Multi-agent approach to dynamic Pick-up and Delivery Problem with uncertain knowledge about future transport demands, Fundamenta Informaticae : Annales Societatis Mathematicae Polonae 71 (2006) 27–36.

[6] J. Kozlak, J.-C. Crput, V. Hilaire, A. Koukam, Multi-agent environment for modelling and solving dynamic transport problems, Computing and Informatics 28 (3) (2009) 277–298, slovak Academy of Sciences. Institute of Informatics.

[7] E. Nawarecki, J. Kozlak, G. Dobrowolski, M. Kisiel-Dorohinicki, Discovery of crises via agent-based simulation of a transportation system, in: Multi-Agent Systems and Applications IV, 4th International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2005, Budapest, Hungary, September 15-17, 2005.

[8] S. Ambroszkiewicz, W. Bartyna, K. Cetnarowicz, M. Faderewski, G. Terlikowski. Interoperability in open heterogeneous multirobot systems. In *Proc. RIDIS2007 Fall 2007 Symposium, Arlington, Virginia USA*, pages 24–31. AAAI, USA, 2007.

[9] S. Ambroszkiewicz and K. Cetnarowicz. On the concept of agent in multi-robot environment. In Springer-Verlag Lecture Notes in Computer Science, Volume 3825/2006, Berlin Heidelberg, 2006.

[10] W. Turek, K. Cetnarowicz, and W. Zaborowski. Software Agent Systems for Improving Performance of Multi-Robot Groups *Fundamenta Informaticae*, 112(1):103–117, 2011.

[11] W. Turek. Extensible Multi-Robot System. In *Computational Science - ICCS 2008, Lecture Notes in Computer Science*, pages 574–583. Springer-Verlag, Berlin, Heidelberg, 2008.

[12] A. Byrski, M. Kisiel-Dorohinicki. Agent-Based Meta-Heuristic Approach to Discrete Optimization. International Conference on Complex, Intelligent and Software Intensive Systems (CISIS), 2011, p. 508–512.